ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.19, No.8, August 2008, pp.2102−2113
DOI: 10.3724/SP.J.1001.2008.02102

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

基于规则推导的特权隐式授权分析*

蔡嘉勇[1,2,5+], 卿斯汉[1,2,3,4], 刘 伟[1,5], 何建波[1,2,5]

[1](中国科学院 软件研究所 基础软件国家工程研究中心,北京 100190)

[2](中国科学院 软件研究所 信息安全技术工程研究中心,北京 100190)

[3](北京大学 软件与微电子学院,北京 102600)

[4](北京中科安胜信息技术有限公司,北京 100086)

[5](中国科学院 研究生院,北京 100049)

Analysis on Implicit Authorization in Privilege Through Rule Deduction

CAI Jia-Yong[1,2,5+], QING Si-Han[1,2,3,4], LIU Wei[1,5], HE Jian-Bo[1,2,5]

[1](National Engineering Research Center for Fundamental Software, Institute of Software, The Chinese Academy of Sciences, Beijing 100190, China)

[2](Engineering Research Center for Information Security Technology, Institute of Software, The Chinese Academy of Sciences, Beijing 100190, China)

[3](School of Software and Microelectronics, Peking University, Beijing 102600, China)

[4](Beijing Zhongke Ansheng Corporation of Information Technology, Beijing 100086, China)

[5](Graduate University, The Chinese Academy of Sciences, Beijing 100049, China)

+ Corresponding author: E-mail: jiayong02@ios.cn

Cai JY, Qing SH, Liu W, He JB. Analysis on implicit authorization in privilege through rule deduction. *Journal of Software*, 2008,19(8):2102−2113. http://www.jos.org.cn/1000-9825/19/2102.htm

Abstract: A scheme on studying the safety issues for privilege in systems is introduced. Since the particular faculty of transiting security states makes analyzing and protecting privilege for a system difficult, techniques used in traditional access control should not copy to this field. For this reason the features are firstly inspected by discussing the origination of privilege using access control space theory. Then rules defined for a system could be divided into two categories: constraint rules and execution rules, describing the restrictions and effect of an authorization respectively. Furthermore, a special authorization relation between different privilege operations, as well as its properties, is investigated against rules' logical patterns by deduction. A quick algorithm for constructing authorization deduction graph is also provided. Basing on it, common safety issue of implicit authorization was reviewed with the possibility to be abused. Finally this paper formalizes the capability mechanism defined by POSIX (portable operating system interface) standard, constructing ADG (authorization deduction graph) for it. The

* Supported by the National Natural Science Foundation of China under Grant No.60573042 (国家自然科学基金); the National Basic Research Program of China under Grant No.G1999035802 (国家重点基础研究发展计划(973)); the Beijing Natural Science Foundation of China under Grant No.4052016 (北京市自然科学基金)

Received 2006-12-06; Accepted 2007-03-26

design is revised with countermeasures against privilege abusing so as to preserve consistent with the principle of least privilege.

Key words: privilege; constraint rule; execution rule; deduction; implicit authorization

摘 要: 介绍了一种研究系统特权安全问题的方法.由于其特有的迁移系统安全状态的能力,使得分析及保护系统特权都很困难,因此,传统访问控制研究中所采用的技术无法复制到该领域.在访问控制空间理论下,检查了系统特权的来源问题及其特点,从而将系统规则划分为约束规则与执行规则两类,分别描述授权的限制与效果.进一步对规则逻辑形式进行推导,发现特权操作间的特殊授权关系以及相关属性,并设计了一种快速构造授权推导图的算法.在此基础上,分析隐式授权安全问题可能存在的滥用特权威胁.最后对 POSIX(portable operating system interface)标准的权能机制进行形式化描述,计算并构造其授权推导图.对标准设计中存在的滥用威胁提供了对策,有效地实现了与最小特权原则的一致性.

关键词: 特权;约束规则;执行规则;推导;隐式授权

中图法分类号: TP309 **文献标识码**: A

访问控制机制是操作系统提供安全保护的基础,针对不同的应用安全目标应制定不同的访问控制策略.而研究者对访问控制策略建立数学分析模型(如 BLP,Biba,DTE,RBAC 等),目的在于验证策略与安全目标的一致性,避免未经授权的主体访问系统资源.然而基于实施的访问控制策略,系统仍然存在执行某种行为的能力,称为特权(privilege)[1].由于其拥有修改系统安全状态的能力,误用、滥用及泄漏特权都将对系统造成极大的风险.为此,Saltzer提出最小特权原则(principle of least privilege)[2]以限制主体可使用的能力范围,它要求“系统中的每个程序和用户应当使用足以完成任务的最小特权集合进行操作”.

针对最小特权原则的有效实施问题,Schneider 认为还需考虑授权度量性问题以及在进程运行时进行特权状态迁移的好处[3].我们认为,当前特权领域的研究可分为 3 类:第 1 类研究特权表示及底层实现,以提供细粒度且不可绕过的访问控制,如 POSIX(portable operating system interface)[4]提出的权能机制,文献[5]则将其权能集扩展为 384 个元素;在此基础上,第 2 类分析进程生命周期的特权需求与触发事件的关系,将进程被许可的能力与程序逻辑密切关联,如基于状态的特权控制模型 SBPC[6];第 3 类考虑特权控制的层次,如文献[1]认为,特权的控制分为 3 个层次,不同层次的最小特权内涵与控制方式不同,通过不同的条件加以限制,逐层缩小主体的特权范围,使其达到最小特权的目标.

从现有研究成果来看,特权研究还存在两个主要问题:首先,最小特权原则描述的安全目标不明确,而特权机制的设计也与具体的系统结构紧密相关,为其建立抽象的数学模型比较困难.由于缺乏安全目标的一致性验证,系统的特权机制无法提供严格的安全保证;其次,现有研究存在仅关注授权约束的缺陷,对权限的能力分析不足,造成管理员无法准确计算授权的能力范围、评估其安全风险,授权配置中最小特权原则的实施主要依赖主观经验进行.

基于访问控制空间理论[7],本文第 1 节比较特权机制与传统访问控制策略在建模方式上的差别与关系,将建立完备特权约束系统分析模型的规则分为两类,其中,约束规则描述特权的授权约束,新增的执行规则描述权限对系统安全状态的影响,以弥补以往研究仅关注授权约束的缺陷.研究模型规则的逻辑形式,可发现系统规则间存在推导关系,进而产生授权推导,由此,第 2 节提出基于规则推导的特权授权分析方法,并设计了一种快速构造授权推导关系图的算法.管理员只需搜索图中能力节点经有向路径的所有可达节点,即可得准确的授权能力边界.而授权推导产生的隐式授权具有隐蔽性,往往为研究者所忽略,导致系统存在不必要的特权滥用风险.基于授权推导关系图,第 3 节探讨滥用隐含产生的主要原因及可能的对策.第 4 节使用 POSIX 权能机制实例化特权约束的系统模型,分析结果显示该机制存在 7 种安全隐患,并给予解决方案.最后总结全文.

1 基于特权的建模

1.1 特权的来源

Jaeger 将策略模型的访问控制空间(access control space)[7]定义为主体从策略中可获得的所有权限与能力(本文将权限与能力视作执行某种操作的许可,除非特别说明,以下能力、权限与操作意义等同).普通安全策略模型(是指除了特权机制模型以外,用于描述系统访问控制策略的其他模型,如 BLP 模型、DTE 模型、RBAC 模型等)的安全目标是防止未经授权的主体访问系统资源,针对应用需求分为机密性、完整性等具体解释.其关心的是访问客体的安全性,模型描述与约束的访问控制空间有限(通常仅包括读、写、执行等客体相关操作).而系统的访问控制空间并不限于此,传统策略模型约束之外能力的执行往往控制系统安全状态的变化(如关机等),应当对其加以约束.由于对这部分能力缺乏明确的安全控制目标,难以在统一的数学模型中完成分析,大多数系统将这类操作标记为特权,赋予特权机制管理,这构成特权来源的第 1 部分.

操作系统引入的访问控制策略模型也带来两个方面的问题:首先,策略过于严格的授权约束导致系统不可用,比如经典 BLP 模型规定系统只存在向上写的信息流.从 MULTICS 到最新的安全操作系统都存在这样的问题,为满足可用性要求,必须提供特殊情况下的超越授权.其次,随着新策略的加入,系统状态空间必然添加新的安全变量以描述其配置状态.而策略可定制系统要求提供对配置状态的修改能力,以满足用户的具体安全需求.对于传统策略,配置状态发生改变意味着具体安全目标的变化,为避免出现控制后门,造成安全一致性验证的复杂化,模型通常不对这部分能力进行描述.加入新访问控制策略而引入的额外能力主要用于满足可用性需求,同样缺乏明确的安全目标,从而构成特权来源的第 2 部分.文献[6]将系统特权分为“超越全局安全策略、进行其他策略无法控制的安全攸关的操作和系统安全策略自身的配置和维护”3 类,从侧面反映出我们对这两种来源的判断.

综上所述,特权机制授予的权限为系统能力的全部,其访问控制空间实际上等于系统的访问控制空间,普通策略的访问控制空间则为其子集.如图 1 所示,访问控制空间交叉程度不同,二者在访问控制决策中的关系也不同:在普通策略的访问控制空间内,特权体现为对其约束的超越关系;在普通策略访问控制空间外,但在其引入的访问控制空间内,特权表现为对策略状态的配置关系.此外,二者则不存在任何关系.因此,特权拥有的能力将直接或间接影响系统的访问控制决策,对系统访问控制的全面安全性分析及验证都不应回避特权问题.

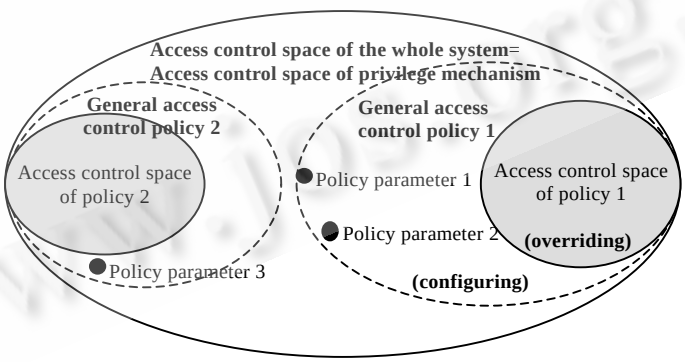

Fig.1 Access control space comparison between privilege and general policy models

图 1 特权与普通策略模型访问控制空间的比较

1.2 约束规则与执行规则

特权建模的困难一方面是由于特权安全目标不明确,另一方面则是由于没有考虑特权机制与传统访问控制的差异,未对其建模方式加以改进.传统策略模型的约束规则主要基于〈主体,客体,操作〉三元组描述主、客体关系.但 Bertino[8]也指出,该模式不足以应付新型数据库及应用模型对访问控制灵活性及表达性的要求.对于访

问客体之外的权限,授权的约束条件不依赖主、客体关系构建.防止未经授权的主体执行系统操作才是特权机制的主要目标.模型规则不应限于三元组描述方式,而应围绕受控能力,灵活指定系统元素之间的关系.

此外,以往访问控制领域的研究,包括特权机制,只关注如何为系统制定严格的授权约束,用以防止未经授权的访问.但主体是否允许执行某种操作,是由策略约束与策略配置共同决定的.约束条件的定义仅能说明配置权限的方法,无法解释授权的原因.关键在于缺乏对权限能力的准确描述,造成管理员无法理解授权的真实内涵,更无法分析与评估所授予的权限范围是否超过了可用性要求.对权限拥有能力的分析不足,导致当前以经验为主导的配置根本不能提供最小特权原则的安全保证.为此,模型中完整提供如下两类规则是非常必要的:

定义 1(约束规则(constraint rules)). 定义系统对能力的所有保护性限制,即主体获得授权必须满足的条件.形式为$\forall X_{1,\ldots,k}\bullet\varphi(X_1,X_2,\ldots,X_k)\Rightarrow op(X_1,X_2,\ldots,X_k)$,其中,前置条件谓词$\varphi(X_1,X_2,\ldots,X_k)$检查当前上下文环境是否满足特定关系,后置状态$op(X_1,X_2,\ldots,X_k)$为受保护的系统能力,形参$X_i(i=1,\ldots,k)$为规则使用的系统变量.本文用 *CR* 表示系统约束规则的全体.

定义 2(执行规则(execution rules)). 描述能力执行对系统安全状态的修改,即对系统的影响.形式为$\forall X_{1,\ldots,k}\bullet op(X_1,X_2,..,X_k)\Rightarrow\theta(X_1,X_2,..,X_k)$,其中,前置条件 $op(X_1,X_2,\ldots,X_k)$为执行的能力,后置状态谓词$\theta(X_1,X_2,\ldots,X_k)$描述该操作执行后对系统安全状态的所有修改.由于并非所有能力都引起系统安全状态变化,因此部分操作不存在对应的执行规则,如 *read*(),*write*()等.本文用 *ER* 表示系统执行规则的全体.

HRU[9]模型采用类似编程语言的建模方法:命令是能力的执行接口,由条件判断与命令体两部分组成,只有满足条件判断的执行才是允许的;命令体的执行是一个原子操作序列,原子操作与描述系统安全状态变化的谓词对应.本文新增的执行规则将权限与描述安全状态变化的谓词直接对应,取消了原子操作的中间映射层次,降低了系统描述的复杂性.通常,执行规则的后置状态可写成简单谓词的合取式$\theta=eff_1\wedge eff_2\wedge\ldots\wedge eff_s,eff_i(i=1,\ldots,s)$描述一种简单效果,表示该能力的执行为一组简单效果的累加,对应于 HRU 中命令体的原子操作序列.这样,执行规则的通用形式为 $op\Rightarrow eff_1\wedge eff_2\wedge\ldots\wedge eff_s$.

第 1.1 节的分析说明,系统的安全保护由普通安全策略与特权机制共同提供,二者缺一不可.鉴于二者在访问控制空间中的关系,执行规则应具有特定形式.特权单独控制能力的约束规则形式较为简单:$pc\Rightarrow op,pc$ 为特权判断谓词.而普通策略访问控制空间内的能力,特权需要超越该策略的约束,其形式较为复杂:规则前置条件应为策略定义的约束条件与特权判断谓词的析取式,即 $Const\vee pc\Rightarrow op$.考虑系统实施多策略的要求,策略的访问控制空间可能重叠,部分能力同时受到多种策略的约束,通常使用一种策略禁止则系统禁止的规则综合其决策结果,写成谓词形式为$(Const_1\vee pc_1)\wedge(Const_2\vee pc_2)\wedge\ldots\wedge(Const_t\vee pc_t)\Rightarrow op$.设策略 i 的约束条件谓词 $Const_i$ 总可以写成一组简单条件判断谓词的析取式 $Const_i=c_{i,1}\vee c_{i,2}\vee\ldots\vee c_{i,n_i}$,可认为模型中约束规则具有通用形式 $(c_{1,1}\vee c_{1,2}\vee\ldots\vee c_{1,n_1}\vee pc_1)\wedge(c_{2,1}\vee c_{2,2}\vee\ldots\vee c_{2,n_2}\vee pc_2)\wedge\ldots\wedge(c_{t,1}\vee c_{t,2}\vee\ldots\vee c_{t,n_t}\vee pc_t)\Rightarrow op$.

使用超级用户身份表示特权的方法,授权粒度太粗,一旦程序需要特权,就必须赋予其所有特权,严重违反最小特权原则,这是目前网络热衷于攻击 root 进程的主要原因.开放操作系统接口标准 POSIX10003.1e 定义了一种新的特权表示方法,系统特权被划分为可指派给用户或主体的单位,称为权能(capability)[4].内核中普遍实施的权能检查一方面提供了可分辨的特权能力,另一方面权能可任意组合支持灵活的指派.权能机制简单明了,且易扩展,成为目前大多数安全操作系统特权的实现方式.本文研究细粒度特权的授权安全问题,因此规定特权判断谓词形式为 $pc{:}priv\in privset(subj)$,其中,*priv* 为特权表示单位,*privset*()获得主体 *subj* 当前拥有的特权集合.

本节划分的两类规则只是对特权约束系统的建模过程提出要求,模型需要定义的变量及规则都必须参照具体分析的特权机制进行实例化.第 4 节及附录给出了对 POSIX 权能机制建模的所有变量及部分规则的定义.以附录中 *chown*()的规则为例,约束规则 RR_1 仅说明主体要拥有修改客体属主关系的能力,必须向其权能集增加 *CAP_CHOWN* 权能标志,却没有解释主体拥有该特权是否恰当、是否存在不受约束而滥用的可能.而执行规则 ER_1 则准确地解释了该特权所拥有的能力.下一节将介绍基于规则推导进行特权授权分析的方法.

2 规则推导与隐式授权

2.1 规则推导

第 1.2 节对模型规则的逻辑形式进行了详细分析,可发现约束规则的前置条件仅包含对系统元素关系的判断,而执行规则的后置状态也只定义了系统安全状态变量的变化,二者可能存在推导关系:

定义 3(规则推导(rule deduction)). 设 $r_1,r_2\in CR\cup ER$,且 r_1:$pre_1\Rightarrow post_1$,r_2:$pre_2\Rightarrow post_2$.若根据谓词形式推演规则 r_1 的后置状态可获得 r_2 前置条件的逻辑推论,即 $post_1\Rightarrow pre_2$,则称 r_1 到 r_2 构成规则推导:$r_1\Rightarrow r_2$.

从其具体含义来看,执行规则修改了系统安全状态变量,导致系统元素之间的关系产生变化,可能影响某一约束规则的访问裁决结果,根据形式推演规则可以证明规则推导具有以下性质:

定理 1. 规则推导关系具有非自反性、非对称性以及传递性.

证明:规则 r 的后置状态 $post$ 未必获得其前置条件 pre 的推论,自反性不成立.设 r_1 与 r_2 构成规则推导,然而 r_2 的后置状态 $post_2$ 未必获得 r_1 前置条件 pre_1 的推论,对称性不成立.设 $r_1\Rightarrow r_2$ 且 $r_2\Rightarrow r_3$,由定义 3 可得 $post_1\Rightarrow pre_2$ 且 $post_2\Rightarrow pre_3$,根据逻辑推论的传递性可得 $post_1\Rightarrow pre_3$,因此 $r_1\Rightarrow r_3$,规则推导具有传递性. □

定理 2. 在特权约束的系统模型中,规则推导关系仅存在于不同类的规则之间.

证明:$r_1\Rightarrow r_2$,设 $r_1,r_2\in CR$,则 r_1 后置状态必须获得 r_2 前置条件的推论,定义 1 指出 r_1 后置状态为受控能力,因为无法直接由其获得表示约束条件判断的谓词的推论,该推导关系不成立.同理,若 $r_1,r_2\in ER$,定义 2 指出描述系统安全状态修改的谓词不可能直接获得受控能力的推论,则该推导关系也不成立.因此,规则推导仅存在于不同类型的规则之间.若 $r_1\in CR,r_2\in ER$,则因前者后置状态与后者前置条件均为受控能力,所以 r_1,r_2 须对应同一能力;反之, $r_1\in ER,r_2\in CR$,若属于同一能力 op,假设管理员需要撤销主体 s 的 op 权限,则 s 可在撤销请求发出与撤销生效之间,通过执行 r_2 修改安全状态变量,调整系统元素关系重新获得授权 op.然而,完整的访问控制机制要求有效的撤销机制,因此,这类推导要求规则对应的能力不同. □

模型规则的划分与定义是为了全面表示系统授权,尤其约束规则解释了主体获得授权的途径,在具体应用中,规则推导关系实际反映的是权限之间的关系,因此可以给出如下定义及相关性质:

定义 4(授权推导(authorization deduction)). 设 $op_1,op_2\in OP$,OP 为模型能力集,$r_1,r_2,r_3,r_4\in CR\cup ER$,其中,$r_1,r_2$ 分别是 op_1 的约束规则与执行规则,r_3,r_4 为 op_2 的约束规则与执行规则.定理 2 指出同一能力的两类规则构成推导,若 r_2 到 r_3 也构成推导:$r_2\Rightarrow r_3$,则可构成更长的规则推导序列 $r_1\Rightarrow r_2\Rightarrow r_3\Rightarrow r_4$,我们称 op_1 到 op_2 构成授权推导 $op_1\Rightarrow op_2$.连续多个授权推导构成的序列称为授权推导序列,序列中不同能力的个数为序列长度.以能力为节点、以授权推导关系为边构造的图称为授权推导关系图(authorization deduction graph,简称 ADG).

推论 1. 授权推导关系具有非自反性、非对称性和传递性.

证明:定理 2 的证明指出,由于访问控制对于撤销机制的要求,同一能力 op 的执行规则无法推出约束规则,根据定义 4,$op\Rightarrow op$ 不满足,授权推导不具有自反性.由于规则推导关系具有非对称性与传递性,根据定义 4,授权推导关系显然也具有非对称性与传递性. □

定理 3. 授权推导关系图 ADG 为有向无环图.

证明:推论 1 说明 ADG 为有向图,因此只需证明图中不存在有向环.授权推导的非自反性说明 ADG 不存在长度为 1 的环.若图中有长度为 n 的环 $cycl$:$op_1\Rightarrow op_2\Rightarrow\ldots\Rightarrow op_n\Rightarrow op_1$($n$>1),那么可同样构造授权撤销失效的场景:当管理员向系统请求撤销主体 s 权限 op_1 时,由于 s 或者其他主体 s' 已获得 op_j(1<j≤n),可调整系统元素关系重新获得授权 op_1.这与完整访问控制的撤销有效性要求矛盾,命题得证. □

由于并非所有能力均存在执行规则,因此$|ER|\le|CR|$.考虑授权推导关系是由执行规则到约束规则的推导关系产生,ADG 中最长的授权推导序列长度不超过$|ER|$.本节建立了基于规则推导的授权分析方法的基础,两类规则的定义弥补了以往研究缺乏特权能力描述的缺陷,而规则与授权间的推导关系则提供了准确计算及评估授权合理性的依据.在权限配置过程中,管理员检查主体拥有权限 op 后系统的安全性,只需遍历 ADG 中节点 op 及其经有向路径所有可达节点,则可获得授权后主体拥有的能力边界.传统建模没有考虑权限对系统的影响,无

法支持对授权能力总体边界的计算,虽然大量研究了授权约束方面的问题,但最小特权原则的实施效果不理想.

2.2 隐式授权定义

Rabitti 等人在讨论下一代安全数据库访问控制模型[10]时提出了隐式授权概念,是指在系统显式授权基础上,由规则或推导可进一步获得的授权,用于缩小显式策略配置的规模[8].如规定未显式指定的授权都拒绝,就是传统访问控制中最常用的隐式授权策略.在其协助下,管理员只需给出所有允许的授权即可,而不必列举对所有访问请求的配置.文献[11]列举了高层访问控制机制 RBAC 的 4 类隐式授权,如对子角色的授权隐含对父角色进行授权、撤销父角色权限表示子角色权限也必须撤销等.针对隐式授权,本文给出如下两条定义:

定义 5(显式授权(explicit authorization)). 若策略配置显式指定了主体 s 拥有执行操作 op 的能力,则称策略对主体 s 存在显式授权 op.

定义 6(隐式授权(implicit authorization)). 策略配置对主体 s 显式授权 op,op'非主体显式授权,若由授权 op 根据授权规则可推导某主体 s'事实上可拥有权限 op',则称策略对主体 s'存在隐式授权 op'.

由第 2.1 节的分析可以看出,与以往分析的隐式授权不同,特权约束的系统中隐式授权规则不是显式指定的,而是蕴涵于模型规则以及授权间的推导关系中,由权限对系统安全状态修改的能力所决定.在 ADG 中,节点的执行规则描述的是能力对系统的直接影响,而可达节点的执行规则描述的是该能力可通过授权规则间接对系统施加的影响,即隐式授权.显式授权与隐式授权的全部,体现了该权限对系统的能力范围.

2.3 弱授权推导关系图构造算法

在实例化特权约束系统模型并计算规则间推导关系时,鉴于规则使用的谓词形式,我们发现完全满足定义 3 要求的复杂谓词间推导关系基本上不可能,原因在于执行规则对系统安全变量状态的修改可能仅调整了部分约束条件的判断结果.但并不是说实际应用中不会出现隐式授权问题,考虑恶意用户完全可能通过正常授权机制满足剩余约束条件,这种可能性的隐式授权才是管理员所关心的.由此需要进一步给出如下定义:

定义 7(弱推导(weak deduction)). $op_1,op_2\in OP,er_1$ 是 op_1 的执行规则,其对系统安全状态的变化仅满足 op_2 约束规则 cr_2 中部分约束条件,假设其他必需的约束条件已满足,那么 r_1 到 r_2 构成规则推导,称为弱规则推导 $r_1\mapsto r_2$,由其进一步获得的授权推导关系称为弱授权推导 $op_1\mapsto op_2$.

对规则间推导构成条件的放宽,使得规则及授权推导关系由必然性转化为可能性.定义 7 仍然基于谓词的形式推演性质,因此,第 2.1 节证明的性质仍然适用于弱授权推导.然而,谓词间的推导计算即使对于简单谓词也是异常困难的,为此,针对不同规则类型构造如下集合,并用定理证明了一种快速构造 ADG 的方法.

定义 8(限制集(restrict set)). 设特权约束的系统模型中约束规则具有通用谓词形式:

$$(c_{1,1}\vee c_{1,2}\vee\ldots\vee c_{1,n_1}\vee pc_1)\wedge(c_{2,1}\vee c_{2,2}\vee\ldots\vee c_{2,n_2}\vee pc_2)\wedge\ldots\wedge(c_{t,1}\vee c_{t,2}\vee\ldots\vee c_{t,n_t}\vee pc_t)\Rightarrow op,$$

称 $(c_{j,1},c_{j,2},\ldots,c_{j,n_j},pc_j)$ 为 op 的一个限制集($1\le j\le t$),用 pc_j 中的特权标志 $priv_x$ 标识该集合 RS_{priv_x}.

定义 9(效果集(effect set)). 执行规则谓词形式为 $op\Rightarrow eff_1\wedge eff_2\wedge\ldots\wedge eff_s$,称$(eff_1,eff_2,\ldots,eff_s)$为 op 的效果集,用操作 op 标识该集合 ES_{op}.

定理 4. op_1 效果集中某元素 $e\in ES_{op_1}$ 对 op_2 一个限制集中的元素 $c\in RS_{priv_x}$ 存在蕴含关系 $e\to c$,当且仅当 op_1 的执行规则与 op_2 的约束规则构成弱规则推导,op_1 与 op_2 构成弱授权推导.

证明:用形式推演定律证明充分性.由$\wedge$消去律可得 $op_1\Rightarrow\theta_1\Rightarrow e$,由$\to$消去律可得 $e\Rightarrow c$,由$\vee$引入律可得 $c\Rightarrow c_1\vee c_2\vee\ldots\vee c_s\vee pc\vee c=Const_m$.由定义 7 可得 $(c_1\vee c_2\vee\ldots\vee c_s\vee pc\vee c)\mapsto(c_1\vee c_2\vee\ldots\vee c_s\vee pc\vee c)\wedge\ldots=\varphi_2$,综上,$\theta_1\mapsto\varphi_2$.执行规则 r_1:$op_1\Rightarrow\theta_1$ 与约束规则 r_2:$\varphi_2\Rightarrow op_2$ 构成弱规则推导,op_1 与 op_2 构成弱授权推导.

必要性证明.设 r_1:$op_1\Rightarrow\theta_1$,r_2:$\varphi_2\Rightarrow op_2$ 构成弱规则推导 $r_1\mapsto r_2$,由定义 7 可知,θ_1 仅需部分满足 φ_2 规定的限制条件,比如 $Const_i=c_{i,1}\vee c_{i,2}\vee\ldots\vee c_{i,n_i}$.因为 $Const_i$ 为合取式,只要存在 $c=c_{i,t}(1\le t\le n_i)$,使得 θ_1 满足 c 即可.由于 $\theta=e_1\wedge e_2\wedge\ldots\wedge e_s$,那么必然存在 e 满足 $c_{i,t}$,根据定义 8、定义 9,$c_{i,t}$ 为 op_1 限制集元素,e 为 op_2 约束集元素.由$\to$引入律可得蕴含关系 $e\to c$. □

由于推导仅存在于不同能力的不同规则之间,定理 4 的证明过程说明,只需依次遍历计算不同能力的限制

集与效果集中简单谓词的蕴含关系,即可获得规则与授权间的弱推导关系.由于简单谓词间的蕴含关系计算简单:赋值谓词蕴含对被赋值变量进行关系判断的谓词,因此大大降低了推导计算难度,由此可构造如下快速算法:

算法. 弱授权推导关系图 ADG 构造算法.

输入:能力集 *OP*,所有限制集总集 $\sum RS$ 与效果集总集 $\sum ES$.

输出:授权推导关系图(*G*,*V*).

const_op(*RS*):返回与限制集 *RS* 对应的操作 *op*.

const_priv(*RS*):返回与限制集 *RS* 对应的特权标志 *priv*.

eff_op(*ES*):返回与效果集 *ES* 对应的操作 *op*.

const_set(*c*):返回简单约束谓词 *c* 所在的限制集 *ES*.

```
begin
    let G=OP
    let V=∅
    foreach op1,op2 in OP and op1≠op2 do   /*遍历搜索系统所有能力对*/
        let TES= ⋃_{eff_op(ES)=op1} ES   /*构造能力 op1 的效果总集*/
        let TRS= ⋃_{const_op(RS)=op2} RS   /*构造能力 op2 的限制条件总集*/
        foreach e in TES and c in TRS do   /*遍历检查不同能力效果与限制条件*/
            if e→c then   /*存在蕴含关系*/
                V=V∪〈op1,op2,const_priv(const_set(c))〉   /*加入边集,用限制集的特权标识标记边*/
                go out   /*开始搜索下一对能力节点*/
            endif
        endfor
    endfor
end
```

算法对能力集中的所有能力对进行检查,遍历其效果集与限制集,因此算法复杂度为 $o(|OP|^2)$.

3 隐式授权的风险及对策

传统隐式授权所存在的主要问题是其授权规则过于通用化,授权粒度太粗以至于与系统显式授权及约束产生冲突,文献[8]正是检查与解决此类冲突.这是针对隐式授权规则可显式指定的情况,那么对于特权约束系统模型必须满足的最小特权原则来说,由于隐式授权方式隐蔽,往往为管理员所忽略,但特权却拥有修改系统安全变量状态的能力,滥用这部分隐式授权的能力将对系统产生极大的威胁.以附录的 *chown*()为例,其字面解释仅提及允许修改客体属主关系,这属于显式授权部分,实际却隐含着客体自主访问策略配置被修改:原属主的权限被转授新用户,即隐式授权部分.恶意用户完全可以修改文件属主以便篡改或阅读重要文件,而后重新将文件属主改回.管理员若没有意识到这点,则很可能认为系统授权配置尚未修改,系统已经阻止恶意用户访问客体.

Yee[12]认为,要满足最小特权原则,系统接口必须可分辨且具有确定性,即用户的操作接口独立且输出结果与期望一致,授权接口的设计也应满足同样的要求.由上面的例子可知,尽管细粒度化的特权机制已经提供了可分辨的授权接口,限制了主体的特权能力,但隐式授权向管理员屏蔽了部分授权信息,特权能力之间可能存在推导关系,造成授权能力范围与管理员期望不一致,向恶意用户提供了危险的滥用可能.基于规则推导构造的弱授权推导关系图为全面搜索隐式授权奠定了基础.

值得注意的是,并非所有隐式授权都具有危害性,可用性才是特权机制的根本设计目标.因此,可用性要求的隐式授权属于正常的系统要求,需要防范的是由于管理员配置失误或者系统对约束条件限制不严格而造成

的滥用可能性.比如,*chown*()修改文件属主的功能是系统自主访问控制机制可用性的基本需求,但对该能力设置的保护性约束条件过于宽松,使得主体可以轻易地获得滥用的权限.

隐式授权滥用风险的产生主要原因有二:首先,管理员未准确计算授权推导关系图,误解授权的能力范围,从而授予主体过多的隐式能力;其次,不合理的系统约束造成授权限制不足,从而制造了隐式授权的滥用机会.针对不同的滥用产生原因,应采用不同的对策:对于前者,管理员必须准确描述特权约束系统模型,计算并构造ADG,检查每条授权配置的实际能力范围(显式授权与隐式授权的全部)是否刚好完成预期任务,删除不必要条目;后者则应检查 ADG 蕴含的每种隐式授权模式,分析其滥用的可能性,向过于宽松的约束规则或系统环境中增加必要的限制,防止该隐式授权的产生.第 4.2 节分析了 POSIX 权能机制所有的隐式授权滥用风险并给出了对策.

4 应用实例

POSIX[4]定义了细粒度特权控制系统必须支持的基本权能集,主流安全操作系统 SELinux,TrustedBSD, Trusted Solaris 及 DG/UX 都对其进行了扩展,以替代早期使用单一超级用户 root 表示特权的方式.本节对基本权能集进行适当的简化,实例化特权约束的系统模型,并对其隐式授权安全性进行分析.

4.1 模型变量

普通安全策略是特权约束系统模型不可或缺的部分,这是特权存在的原因之一.为避免涉及过多安全策略,本文假设仅实施两种常见的访问控制:自主访问控制(discretion access control,简称 DAC)与多级别安全(multi-level security,简称 MLS)机制.DAC 实施访问控制列表(access control list,简称 ACL)模型,MLS 则实施 BLP 模型:简单安全属性与*-安全属性,即低级别主体不能读高级别客体,高级别主体不允许写低级别客体.POSIX 要求的基本权能集共有 22 种权能,考虑审计不属于访问控制范畴,可以忽略审计相关权能的描述.ACL 需表示用户与组的概念,本文直接简化为属主与非属主两类.因此,除了 *CAP_SETGID* 以及 *CAP_INF_**以外,特权约束的系统模型中共有 15 种权能标志.我们定义系统变量类型:主动实体为主体(即代表用户执行的进程),变量类型为[*PROC*];被动实体为客体、操作及安全策略标签(即权能、安全等级、用户),变量类型为[*FILE*,*OPERATION*, *CAPABILITIES*,*LABEL*,*USER*].模型变量定义如下:

(1) 主体集 S:$\wp PROC$,集合元素是权限的执行者,代表其属主完成任务.

(2) 客体集 O:$\wp FILE$,为普通策略的访问控制对象,是读、写、执行等操作的参数之一.

(3) 操作集 OP:$\wp OPERATION$,系统提供的操作总集,即系统与特权机制的访问控制空间.

(4) 访问操作集 A:(*read*,*write*,*execute*),表示只读、只写、执行,是普通安全策略模型的访问控制空间.

(5) 系统判定集 D:{*yes*,*no*,*undefined*},表示访问裁决结果,分别为策略对主体执行操作许可、禁止与未定义.

(6) 用户集 U:$\wp USER$,是现实用户与系统的接口,主体是其使用系统能力的代理.传统建模方法将用户与进程统一抽象为主体,甚至视作客体的子集,不满足 ACL 的描述要求,本文将用户视作主/客体的 DAC 安全标签.

(7) 安全等级标签集 L:$\wp LABEL$,集合元素表示主/客体的 MLS 安全标签,不同的安全级别标签存在的可能支配关系$\prec$.

(8) 权能集 C:$\wp CAPABILITIES$,集合元素为简化后的 15 种权能.

第 1 节提到,引入安全策略将增加新的状态变量,传统安全策略在建模时通常仅用一个访问控制矩阵表示系统配置状态,无法区分不同安全策略的配置状态,需要加以分解.

(9) 自主访问控制矩阵 $Matrix\subseteq(U\times O\times(2^A\times 2^A))$,定义 DAC 的配置状态,为表达授权与用户标签的关系,矩阵元素首向量使用用户标签,而非主体;末向量为二元组,分别为主体属主等于或不等于客体属主的授权结果.

(10) 安全级别指派 $LA\subseteq((S\cup O)\times L)$,是指 MLS 对主体或者客体的安全标签指派情况.策略根据主体与客体

等级标签的支配关系,决定访问客体的操作是否允许.

(11) 权能许可集 $PA\subseteq((S\cup O)\times 2^C)$,定义主、客体的特权配置状态,只有主体的权能配置状态被用于限制规则的特权判断,客体的权能状态仅用于计算并产生主体权能.

(12) 客体属主函数 $f_o:O\to U$ 获得指定客体的属主标签,这是多对一的映射关系,多个客体可属于同一用户.

(13) 客体访问属性主函数 $f_A:O\to 2^A$ 定义当属主用户访问时,DAC 安全策略的裁决.

(14) 客体访问属性副函数 $f_a:O\to 2^A$ 为非属主用户访问时,DAC 安全策略的裁决.

(15) 客体敏感级(sensitive)函数 $f_m:O\to L$ 获得客体被指派的等级标签.本文不考虑级别范围的表示方法.

(16) 客体特权函数 $f_p:O\to 2^C$ 获得指定客体被指派的权能标记集合.

(17) 主体属主函数 $s_o:S\to U$ 获得指定主体的属主标签,这是多对一的关系,一个用户可拥有多个主体进程.

(18) 主体的清除级(clearance)函数 $s_m:S\to L$ 获得主体的安全级别标签.本文没有采用级别范围的表示方法.

(19) 主体特权函数 $s_p:S\to 2^C$ 获得主体的权能集,新创建主体的权能集由主体与客体的权能集经遗传公式获得.复杂的状态迁移计算不是本文的研究重点,因此,简化规定主体的权能集为原主体权能集与可执行文件权能集的合集,即 $s'=exec(s,o)\Rightarrow s_p(s')=s_p(s)\cup f_p(o)$.

本文将特权约束的系统定义为 $\sigma=(S,O,U,A,L,Matrix,LA,PA,f_o,f_A,f_a,f_m,f_p,s_o,s_m,s_p)$.

4.2 权能机制的安全性分析

在第 4.1 节的实例化模型中,系统一共可授予主体 12 种操作权限,从而构成特权的访问控制空间(见表 1).

Table 1 Explanation of operations in access control space of privileged system model

表 1 特权约束系统模型访问控制空间的操作解释

Operation	Explanation	Operation	Explanation
chown(*s*,*o*,*u*)	Change the owner of object *o* to *u*	*execute*(*s*,*o*)	Execute application object *o*
write(*s*,*o*)	Write to object *o*	*read*(*s*,*o*)	Read object *o*
chacl(*s*,*o*,*a*)	Modify acl of object *o* to *a*	*delete*(*s*,*o*)	Delete object *o*
setuid(*s*,*u*)	Set owner of subject *s* to *u*	*kill*(*s*,*s*′)	Subject *s* send messge to subject *s*′
link(*s*,*o*,*o*′)	Make a link from *o* to *o*′	*setfcap*(*s*,*o*,*c*)	Set capability set of object *o* to *c*
setfmls(*s*,*o*,*l*)	Set security level of object *o* to *l*	*setsmls*(*s*,*s*′,*l*)	Set security level of subject *s* to *l*

ACL 与 BLP 策略的访问控制空间为读、写、执行(部分改进策略模型可拥有更大的访问控制空间).应用第 2.3 节算法构造的 ADG 如图 2 所示,限于篇幅,仅显示存在授权推导关系的能力节点,边用权能标记(省略 *CAP_**前缀).按照有向路径的先后,ADG 可以分解为 6 个树型子图,其中 setfcap()由于与其他操作的直接授权推导关系而单独构成树型子图.图 2 构造使管理员拥有准确计算与评估授权后主体权限边界的能力:主体 *s* 实际能力范围为显式权限 *op* 及其经有向路径所有可达节点的执行规则描述总和.对应于系统安全设计目标,可逐一检查系统的授权结果是否满足最小特权原则的要求,即授权的能力刚好满足可用性要求.可发现 POSIX 权能机制存在以下问题:

(1) 子图①存在环,可能导致对 *setfcap*()的授权无法真正撤销.因此,要求任何时候拥有 *CAP_SETFCAP* 的主体在全系统不得多于 1 个,且禁止 *setfcap*()继续授予任何主体权能 *CAP_SETFCAP*.

(2) 主体 *s* 执行 *setfcap*()为其他需要特权的主体提供特权是系统可用性的要求,但主体可滥用其隐式授权能力为自己增加特权.我们为 *setfcap*()的约束规则增加变量约束谓词,禁止操作 *setfcap*()的主体参数为 *s* 自身.

(3) 子图②说明 *chown*()存在对 *chacl*()及 *delete*()的隐式授权,主体可进一步隐式获得对文件的任意访问.鉴于系统创建的客体将自动赋予主体的属主标签,因此可以不需要该特权.因此,*chown*()隐式授予的能力制造了滥用机会.附录给出了对其授权约束规则增加变量约束的过程:禁止将客体的属主设置为主体 *s* 自身.

(4) *fsetid*()同样隐式地获得对客体的任意访问能力,如子图③所示.然而,这属于系统可用性要求:使应用程序拥有可执行文件客体属主的访问权限.其滥用风险无法使用谓词约束的方法消除,只能要求安全管

理员谨慎分配*CAP_FSETID*权能,并定期检查系统范围所有带*setuid*标志的文件,通常,系统安装后该能力则是多余的,可删除.

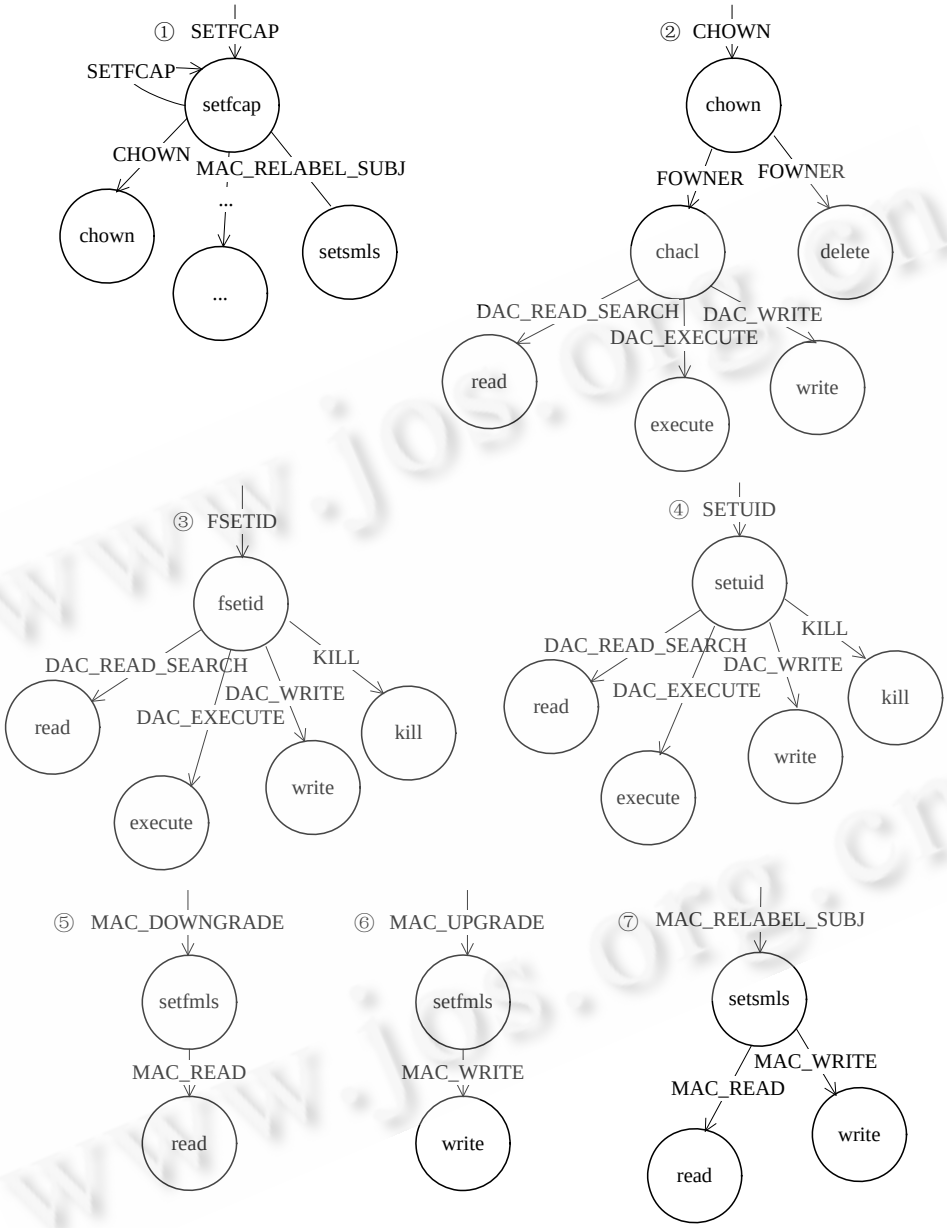

Fig.2 ADG for privileged system model

图 2 特权约束系统模型 ADG

(5) 从子图④可见*setuid*()的能力类似于*fsetid*(),主要用于*login*,*ssh*等登录类程序切换用户身份或超级用户进程降低特权级别,因此,策略可以指定*setuid*()允许切换的身份范围,只有在该范围的切换才是允许的.

(6) 子图⑤、子图⑥显示,拥有*setfmls*()能力的主体实际拥有访问敏感数据的能力,该接口为系统提供了向下的信息流,避免单一信息流造成系统的不可用.然而,缺乏足够的限制,恶意主体可利用其隐式授予的能力肆意访问敏感信息.我们要求,只有最低清除级的主体可拥有 *CAP_DOWNGRADE* 权能标记,最高

清除级的主体允许拥有 *CAP_UPGRADE*,以确保主体自身除了调整客体敏感级别,并不能真正访问客体;

(7) 在子图⑦中,*setsmls*()直接调整主体的清除级别,以便其访问特定敏感级别的信息,这是系统可用性的要求,唯一需要避免的是主体随意修改自身的清除级别以制造滥用机会,可以采用增加约束规则的条件来实现.

此外,ADG 所示的隐式授权属于系统正常可用性需求,没有超出系统正常工作的能力需求范围.

5 总 结

既防止用户滥用系统特权又保证系统可用性目标向来是安全操作系统设计与开发的难点.传统研究局限于授权约束的研究方法,无法实际指导管理员的配置工作,对于最小特权原则的实施也无法提供授权结果与授权目标一致性检验.因此,本文提出约束规则与执行规则描述模型的方法,一方面以数学形式提供权限执行内容的严格描述,另一方面基于规则的推导可以进行授权推导,构造授权推导关系图,进而得以计算授权对系统的影响范围,检查其中可能的安全风险,极大地弥补了现有研究的缺陷.

本文针对复杂谓词之间推导关系计算困难的问题,应用分解法定义了权限的限制集与效果集,使用简单谓词之间的蕴涵关系判断复杂谓词乃至规则、授权间的推导关系,实现了快速的弱授权推导关系图构造算法.授权间的推导导致系统可能出现隐式授权问题,由于其授权方式隐蔽,对系统将造成极大危害.授权推导关系图为全面搜索及消除隐式授权产生的安全隐患奠定了基础.

正如文献[1]所指出的,系统授权控制可以分为管理层、功能层及执行层.实现全面、系统的最小特权原则要求根据 3 个层次的不同内涵分别进行授权安全性分析.本文基于规则推导分析特权的隐式授权问题仅完成了执行层的工作,未来还需要应用该方法分析其他层次的最小特权原则问题.

致谢 感谢中国科学院软件研究所贺也平研究员对本文提出的宝贵建议,也感谢匿名专家对本文研究意义的充分肯定.

References:

[1] Ji QG, Qing SH, He YP. A new formal model for privilege control with supporting POSIX capability mechanism. Science in China (Series E), 2004,34(6):683−700 (in Chinese with English abstract).

[2] Saltzer JH, Schroeder MD. The protection of information in computer systems. In: Trew JR, Calder J, eds. Proc. of the IEEE, Vol.63. New York: IEEE, Inc., 1975. 1278−1308.

[3] Schneider FB. Least privilege and more. IEEE Security& Privacy, 2003,1(5):55−59.

[4] Portable Applications Standards Committee of the IEEE Computer Society. Standards Project, Draft Standard for Information Technology-Portable Operating System Interface (POSIX), PSSG Draft 17. New York: IEEE, Inc., 1997.

[5] Data General. Managing security on DG/UX system. Manual 093701138-09, Westboro: Data General, A Division of EMC Corporation, 2001.

[6] Liang B. Research on trusted process mechanism and related problems [Ph.D. Thesis]. Beijing: Institute of Software, the Chinese Academy of Sciences, 2004. 31−101 (in Chinese with English abstract).

[7] Jaeger T, Zhang XL. Policy management using access control spaces. ACM Trans. on Information and System Security, 2003,6(3): 327−364.

[8] Bertino E, Catania B, Ferrari E, Perlasca P. A logical framework for reasoning about access control models. ACM Trans. on Information and System Security, 2003,6(1):71−127.

[9] Harrison MA, Ruzzo WL, Ullman JD. Protection in operating systems. Communications of the ACM, 1976,19(8):461−471.

[10] Rabbiti F, Bertino E, Kim W, Woelk D. A model of authorization for next-generation database systems. ACM Trans. of Database Systems, 1991,16(1):88−131.

[11] Essmayr W, Kastner F, Pernul G, Preishuber S, Tjoa AM. Authorization and access control in IRO-DB. In: Stanley Y, Su W, eds.

Proc. of the 12th Int'l Conf. Washington: IEEE Computer Society, 1996. 40−47.

[12] Yee KP. User interaction design for secure systems. In: Deng R, Qing S, Bao F, Zhou JY, eds. Proc. of the 4th Int'l Conf. on Information and Communications Security. LNCS 2513, London: Springer-Verlag, 2002. 278−290.

附中文参考文献:

[1] 季庆光,卿斯汉,贺也平.支持 POSIX 权能机制的一个新的特权控制的形式模型.中国科学(E 辑),2004,34(6):683−700.

[6] 梁彬.可信进程机制及相关问题研究[博士学位论文].北京:中国科学院软件研究所,2004.31−101.

附录. 部分权能规则实例计算及修正过程.

以*chown*(*s*,*o*,*u*)与*write*(*s*,*o*)**为例,简单介绍第 4 节对模型规则实例化的过程.相关规则的文字定义参见文献[4]对权能的描述,其中,*write*(*s*,*o*)没有影响系统安全状态,属于原子操作,因此没有与之对应的执行规则:

规则 RR_1:$\forall u:U,o:O,s:S \bullet CAP\_CHOWN \in s_c(s) \Rightarrow chown(s,o,u)$.

规则 ER_1:$\forall u:U,o:O,s:S \bullet chown(s,o,u) \Rightarrow f_o(o)=u$.

规则 RR_2:

$$\forall s:S,o:O \bullet ((s_o(s) \equiv f_o(o) \wedge write \in f_A(o)) \vee (s_o(s) \neq f_o(o) \wedge write \in f_a(o)) \vee CAP\_DAC\_WRITE \in s_c(s)) \wedge (o_m(s) \succ s_m(o) \vee CAP\_MAC\_WRITE \in s_c(s)) \Rightarrow write(s,o).$$

"≡"表示比较,"="则表示赋值,必须加以区分.

根据定义 6、定义 7,可以给出能力的限制集与效果集如下:

$RS_{CAP\_CHOWN}=(CAP\_CHOWN \in s_p(s))$;

$ES_{chown}=(f_o(o)=u)$;

$RS_{CAP\_DAC\_WRITE}=((s_o(s)\equiv f_o(o) \wedge write \in f_A(o)),(s_o(s)\neq f_o(o) \wedge write \in f_a(o)),CAP\_DAC\_WRITE \in s_c(s))$;

$RS_{CAP\_MAC\_WRITE}=(f_m(s) \succ s_m(o),CAP\_MAC\_WRITE \in s_p(s))$.

利用第 2.3 节的算法遍历 *chown*()与 *write*()所有限制集与效果集元素可发现,赋值操作 $f_o(o)=u$ 未对允许设置的属主用户进行限制,蕴含着 $s_o(s)\equiv f_o(o)$关系得到满足,即蕴含关系$(f_o(o)=u)\rightarrow(s_o(s)\equiv f_o(o))$成立.根据定理 4,两个规则之间构成弱规则推导:$RR_1 \mapsto ER_1$.通过与系统特权的可用性目标进行比较,我们发现存在滥用的可能:只要将用户属主设为自己,则拥有写该文件的能力.因此,为 RR_1 增加变量约束谓词,要求主体不可利用该特权修改文件客体属主为自身,修正后的规则如下:

规则 RR_1':$\forall u:U,o:O,s:S \bullet u \neq s_o(s) \wedge CAP\_CHOWN \in s_c(s) \Rightarrow chown(s,o,u)$.

蔡嘉勇(1978−),男,福建莆田人,博士,主要研究领域为信息系统安全理论和技术.

刘伟(1979−),男,博士,主要研究领域为操作系统安全,网络安全.

卿斯汉(1939−),男,研究员,博士生导师,CCF 高级会员,主要研究领域为信息系统安全理论和技术.

何建波(1978−),男,博士,主要研究领域为信息系统安全理论和技术.

** 可以省略与安全分析无关的参数,以降低复杂性.由于 *write*(*s*,*o*,*buffer*,*length*)写入数据的安全性已经由系统访问客体进行了保护,因此后两个参数与本文分析目标无关,予以省略.类似的还有 *read*(*s*,*o*,*buffer*,*length*)等接口.